REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE:**	**3. DATES COVERED** *(From - To)*
23-08-2007	Conference Paper	13 April 2007

4. TITLE AND SUBTITLE

Application of evolutionary algorithms and neural network concepts to the design of low-cost wideband antenna arrays

5a. CONTRACT NUMBER
In-House

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)

Scott G. Santarelli, Robert J. Mailloux, Tian-Li Yu, Thomas M. Roberts, Michelle H. Champion, David E. Goldberg

5d. PROJECT NUMBER
2304

5e. TASK NUMBER
HA

5f. WORK UNIT NUMBER
01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

AFRL/SNHA
80 Scott Drive
Hanscom AFB MA 01731-2909

8. PERFORMING ORGANIZATION REPORT

N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Electromagnetics Technology Division Source Code: 437490
Sensors Directorate
Air Force Research Laboratory
80 Scott Drive
Hanscom AFB MA 01731-2909

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL-SN-HS

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-SN-HS-TP-2007-0010

12. DISTRIBUTION / AVAILABILITY STATEMENT
Statement A: Approved for Public Release; distribution unlimited.

13. SUPPLEMENTARY NOTES
 Presented at 2007 SPIE Defense and Security Symposium, Orlando, FL, 9 – 13 April 2007. One or more of the authors is a U.S. Government employee working within the scope of their Government job; therefore, the U.S. Government is joint owner of the work and has the right to copy, distribute, and use the work. This work is supported by AFOSR; ESC/PA clearance: ESC 07-0394

14. ABSTRACT

This paper describes the application of biologically-inspired algorithms and concepts to the design of wideband antenna arrays. In particular, we address two specific design problems. The first involves the design of a constrained-feed network for a Rotman-lens beamformer. We implemented two evolutionary optimization (EO) approaches, namely a simple genetic algorithm (SGA) and a competent genetic algorithm. We conducted simulations based on experimental data, which effectively demonstrate that the competent GA outperforms the SGA (i.e., finds a better design solution) as the objective function becomes less specific and more "general." The second design problem involves the implementation of polyomino-shaped subarrays for sidelobe suppression of large, wideband planar arrays. We use a modified screen-saver code to generate random polyomino tilings. A separate code assigns array values to each element of the tiling (i.e., amplitude, phase, time delay, etc.) and computes the corresponding far-field radiation pattern. In order to conduct a statistical analysis of pattern characteristics vs. tiling geometry, we needed a way to measure the "similarity" between two arbitrary tilings to ensure that our sampling of the tiling space was somewhat uniformly distributed. We ultimately borrowed a concept from neural network theory, which we refer to as the "dot-product metric," to effectively categorize tilings based on their degree of similarity.

15. SUBJECT TERMS
Genetic algorithm, competent genetic algorithm, optimization algorithm, classification, wideband array

16. SECURITY CLASSIFICATION OF:			**17.LIMITATION OF ABSTRACT**	**18.NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Scott G. Santarelli
Unclassified	Unclassified	Unclassified	UU	13	***19b. TELEPHONE NUMBER*** *(include area code)* N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

Application of evolutionary algorithms and neural network concepts to the design of low-cost, wideband antenna arrays

Scott G. Santarelli[*a], Robert J. Mailloux[b], Tian-Li Yu[c], Thomas M. Roberts[a], Michelle H. Champion[a], David E. Goldberg[c]

[a]Air Force Research Laboratory, Sensors Directorate, 80 Scott Drive, Hanscom AFB, MA, USA, 01731
[b]University of Massachusetts, Department of Electrical and Computer Engineering, 80 Scott Drive, Hanscom AFB, MA, USA, 01731
[c]Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, 104 S. Mathews Ave., Urbana, IL, USA, 61801

ABSTRACT

This paper describes the application of biologically-inspired algorithms and concepts to the design of wideband antenna arrays. In particular, we address two specific design problems. The first involves the design of a constrained-feed network for a Rotman-lens beamformer. We implemented two evolutionary optimization (EO) approaches, namely a simple genetic algorithm (SGA) and a competent genetic algorithm. We conducted simulations based on experimental data, which effectively demonstrate that the competent GA outperforms the SGA (i.e., finds a better design solution) as the objective function becomes less specific and more "general." The second design problem involves the implementation of polyomino-shaped subarrays for sidelobe suppression of large, wideband planar arrays. We use a modified screen-saver code to generate random polyomino tilings. A separate code assigns array values to each element of the tiling (i.e., amplitude, phase, time delay, etc.) and computes the corresponding far-field radiation pattern. In order to conduct a statistical analysis of pattern characteristics vs. tiling geometry, we needed a way to measure the "similarity" between two arbitrary tilings to ensure that our sampling of the tiling space was somewhat uniformly distributed. We ultimately borrowed a concept from neural network theory, which we refer to as the "dot-product metric," to effectively categorize tilings based on their degree of similarity.

Keywords: genetic algorithm, competent genetic algorithm, optimization algorithm, classification, wideband array

1. INTRODUCTION

This paper describes the application of biologically-inspired algorithms and concepts to the design of wideband antenna arrays. In particular, we address two specific design problems. Section 2 describes the application of two evolutionary optimization algorithms to the design of a constrained-feed network for a Rotman-lens beamformer. Section 3 describes a method to measure the "degree of similarity" between two arbitrary planar array geometries. Basically, we borrow a concept from neural network theory, which we refer to as the "dot-product metric," to effectively categorize polyomino arrays based on their degree of similarity. Section 4 contains a summary of our results and conclusions.

2. SIMPLE GA VS. COMPETENT GA: OPTIMIZATION OF A CONSTRAINED-FEED NETWORK

This section directly compares the performance characteristics of simple and competent genetic algorithms (GAs) by applying both methods to the same optimization problem. Section 2.1 provides an overview of the design theory and principles necessary for the invention and implementation of fast, scalable (i.e., competent) genetic algorithms. We briefly outline the specific competent GA used in this experiment, namely the hierarchical Bayesian optimization algorithm (hBOA). Section 2.2 provides an overview of the optimization problem, which involves the design of a

constrained-feed network for a linear array. We discuss the details of each approach (SGA vs. hBOA), and we describe three variations of the objective function used in this experiment. We present the results, demonstrating that the problem is sufficiently difficult such that acceptable solutions are not obtainable using a SGA (regardless of the objective function). On the other hand, we show that the competent GA is able to obtain an acceptable solution when the third objective function is implemented. Finally, in Section 2.3, we summarize our results and draw some interesting conclusions concerning the fundamental differences between simple and competent GAs.

2.1 GA design theory and competent GAs

SGAs are known for their robust optimization capabilities and have been successfully used in a variety of practical antenna problems [1]. A SGA, however, does not always address the issue of *linkage* (*i.e.*, chromosomal encoding) adequately. This section gives an introduction to competent GAs – advanced optimization techniques designed to determine the optimal linkage of bounded, difficult problems.

To better understand the concept of bounded, difficult problems, let us first consider two extreme cases – *OneMax* and *needle-in-a-haystack* (*NIAH*). The objective of the OneMax problem is to maximize the number of ones in a binary string. For example, the optimal solution for a 5-bit chromosome is [1 1 1 1 1]. Furthermore, the fitness of a chromosome is directly proportional to the number of ones in the string. Thus, [0 1 0 1 1], which has a total of three ones, has a higher fitness than [1 1 0 0 0], which only has two ones. The NIAH problem, on the other hand, is defined such that the fitness of one specific chromosome is highest and those of all other chromosomes are equally low. For example, suppose chromosome [1 0 0 1 1] has a fitness equal to 1.0 and chromosome [* * * * *] $\neq$ [1 0 0 1 1] (i.e., every other chromosome) has a fitness equal to 0.0. Hence, chromosome [1 0 0 1 1] is "the needle in a haystack" the algorithm is trying to find.

On one hand, the genes in the OneMax problem are independent with respect to each other, and hence the OneMax problem is fully decomposable. It is considered to be a GA-simple problem, and a SGA can easily solve it within a sub-quadratic number of function evaluations. On the other hand, the order of linkage in the NIAH problem is equal to the chromosome length, and hence the NIAH problem is not decomposable. The NIAH problem is considered to be GA-difficult, and it has been shown that no algorithm can do any better than a random search for this type of problem. On average, it requires an exponential number of function evaluations to find the optimal solution. So we ask ourselves, how efficiently can we solve a problem that falls somewhere between these two extremes? For example, suppose that we have an additive NIAH problem, which is merely a concatenation of several bounded-order NIAHs. Can we find the optimum quickly? More generally, can we design GAs that solve such nearly decomposable problems quickly, reliably, and accurately? In the following sections, we develop an approach that addresses these issues.

2.1.1 The Bayesian Optimization Algorithm (BOA)

For a given problem, the Bayesian optimization algorithm (BOA) [2,3] evolves a population of candidate solutions (i.e., chromosomes) by building and sampling Bayesian networks [4]. BOA first generates a population of candidate solutions either randomly or according to some prior knowledge of the given problem. Then, the population is updated each generation via the following four steps:

(1) Promising candidate solutions are selected using a GA-selection operator such as tournament selection or truncation selection.
(2) A Bayesian network is built to estimate the probability distribution of these promising candidate solutions.
(3) New candidate solutions are generated by sampling the Bayesian network.
(4) The next-generation population incorporates the new candidate solutions by replacing part or all of the current population.

These steps are repeated until some termination criteria are met (i.e., maximum number of generations has been reached, etc.).

2.1.2 Learning Bayesian networks

This section briefly describes how BOA constructs Bayesian networks during evolution. A Bayesian network describes the following Bayesian, joint-probability distribution,

$$p(X) = \prod_{i=1}^{n} p(X_i \mid \Omega_i), \tag{1}$$

where $X = (X_1, X_2, \ldots, X_n)$ is a vector of variables, Ω_i is a subset of X, and $p(X_i \mid \Omega_i)$ is the conditional probability of X_i given Ω_i.

We can visualize a Bayesian network as a directed, acyclic graph, where the nodes represent variables and the edges represent conditional dependencies. For instance, the following Bayesian joint probability can be expressed by the graph shown in Figure 1,

$$p(A,B,C,D,E) = p(A)\,p(B)\,p(C \mid A)\,p(D \mid A,B)\,p(E \mid C,D). \tag{2}$$

Developing a Bayesian network for a specific problem involves two subtasks – learning the network structure, and calculating the conditional probabilities. Calculating the conditional probabilities for a given structure is straightforward, because the probability of each variable in the population is specified. Thus, the maximum likelihood of the conditional probabilities can be obtained by simply calculating the relative frequencies observed in the population. To learn the structure, the current version of BOA adopts a minimum-description-length (MDL) scoring metric in conjunction with a greedy algorithm. Initially, the structure consists of nodes with no edges. The greedy algorithm then updates the graph by one of the following three operators: (1) edge addition, (2) edge removal, or (3) edge reversal. These steps are performed edge by edge under the guidance of the MDL scoring metric, which has two objectives, namely model compactness and model accuracy. A simple network, which does not describe the given data well, possesses high model compactness but low model accuracy; on the other hand, a complex network, which almost perfectly describes the given data, possesses high model accuracy but low model compactness. Ultimately, we desire a network that is somewhere between these two extremes. The greedy algorithm terminates when no improvement to the MDL scoring metric can be made.

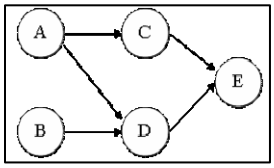

Figure 1. Graphical representation of a Bayesian network.

2.1.3 Hierarchical decomposition – from BOA to hBOA

Hierarchical structures appear in real-world systems and real-world problems [5] and typically refer to situations in which higher-level interactions are not revealed until lower-level interactions have been recognized. We humans often utilize hierarchical decomposition to solve problems in either a bottom-up or top-down manner. Hierarchical decomposition effectively reduces problem difficulty, and hence enables us to solve more difficult problems.

Pelikan [2] recognizes three key issues regarding the success of hierarchical decomposition:

(1) **Proper decomposition.** At each level, the algorithm needs to be capable of properly decomposing the problem. A proper decomposition reduces the problem complexity and hence improves the scalability of the algorithm.
(2) **Chunking.** Each sub-solution in a lower level can be seen as a chunk. The algorithm should be capable of properly representing each chunk as one single variable when solving the next upper level.
(3) **Preservation of alternative candidate solutions.** Since the interactions at a higher level do not reveal themselves until the interactions at the lower levels have been recognized, preserving alternative candidate solutions is important for a hierarchical problem solver.

By applying these three ideas, the hierarchical Bayesian optimization algorithm (hBOA) was constructed. Basically, hBOA utilizes BOA to decompose the problem at successive levels of problem difficulty.

To accomplish chunking, we must consider two objectives. First, the model must be able to merge chunks from lower levels (sub-solutions) to form larger chunks (more complete solutions) at higher levels. Secondly, the model needs to compress the representations of the chunks such that only relevant information is stored. The Bayesian network merges variables into groups implicitly, and hBOA represents sub-solutions efficiently by recognizing the local structures in the Bayesian networks. Specifically, for each variable in the network, hBOA utilizes one decision graph to encode the conditional probabilities associated with the variable. The use of decision graphs reduces the number of conditional probabilities that need to be stored, and hence more complex models can be represented with the same amount of memory space. As a result, more efficient learning is achieved.

In order to preserve alternate candidate solutions, hBOA employs a niching technique, namely *restricted tournament replacement (RTR)* [2, 6]. HBOA has been shown to have a stronger ability than BOA to conquer hierarchical difficulties [2, 6, 7]. A more thorough explanation of hBOA is available in Pelikan's thesis [2] or other publications [6 – 9].

2.2 Problem statement: optimization of a constrained-feed network for a linear array

This section describes an antenna system designed for space-based and airborne radar applications. The goal of this system is to produce a far-field radiation pattern having at least –30 dB sidelobes over a 20% frequency bandwidth. This is accomplished by implementing an optimized, constrained-feed network. The following overview is intended to provide the reader with enough background information to understand the details of the system optimization. For further information about the system design and implementation, the reader is referred to [10].

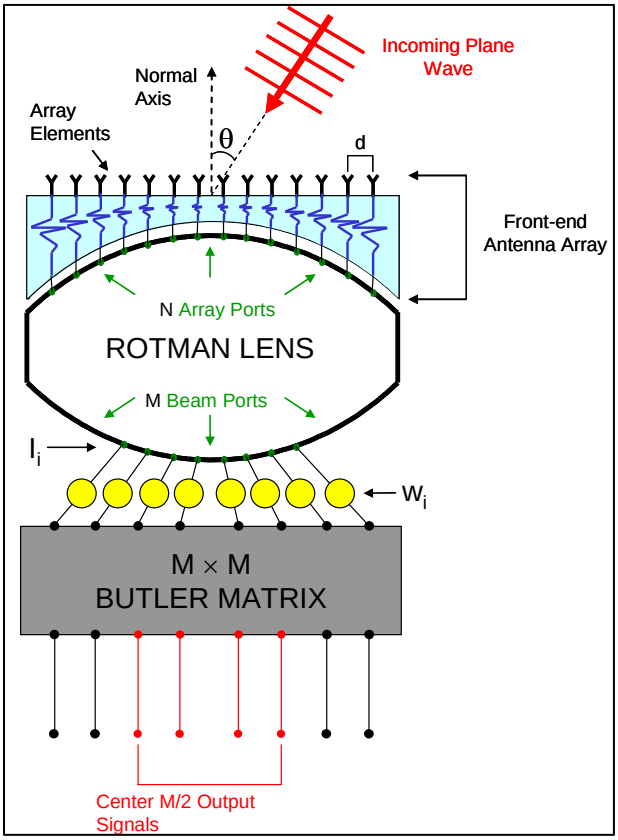

Figure 2. Section block diagram.

Figure 2 shows a single section of the antenna system, which basically consists of a front-end array and a constrained feed network. An incoming plane wave impinges the *N*-element linear array, and the resulting element excitations are propagated through an *N* by *M* Rotman lens, the outputs of which are weighted and fed into an *M* by *M* Butler Matrix. The center *M*/2 Butler outputs from each of *P* sections are time-delayed, weighted, and combined to compute the final radiation pattern of the system. In order to meet the system specifications described above, we need to optimize the weights that lie between the Rotman lens beamports and the Butler matrix inputs (i.e., w_i) for *P* sections of the system.

The system parameters are as follows: frequency band of operation = 9.0 – 11.0 GHz, center frequency f_0 = 10.0 GHz, N = 64, $d = 0.5\lambda_0$ (center-frequency wavelength), M = 8, P = 3.

2.2.1 Simple genetic algorithm (SGA)

The antenna system consists of three sections, each containing eight complex weights (*i.e.*, amplitude and phase); therefore, we must optimize a total of 24 complex weights in order to meet the system requirements. Figure 3 shows the chromosomal representation. We chose a binary encoding scheme, and we represented each complex weight with 16 bits (i.e., 8 bits amplitude and phase). Thus, the length of the chromosome is 384

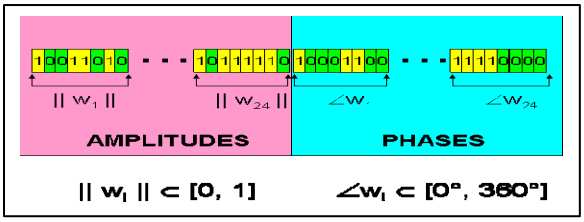

Figure 3. Chromosomal representation

bits. We arbitrarily chose to encode the amplitudes along the first half of the chromosome and the phases along the second half. Both the amplitudes and phases are numbered sequentially along each half as shown in the figure. We restricted the amplitudes to lie in the interval [0 1] and the phases to lie in the interval [0° 360°]. It is also worth noting that we used an 8-bit gray code for both the amplitude and phase encoding schemes [1].

Initially, the algorithm forms a random population of 200 chromosomes (i.e., parents). Each member of the population is evaluated and ranked by the objective function. Next, a mating pool of 200 individuals is created via binary tournament selection (*n*-tournament selection is when *n* individuals are randomly selected from the population, and the one with the highest fitness is selected for the mating pool [11]). Next, two individuals from the mating pool are chosen randomly to create an offspring via two-point crossover. This process is repeated until 200 offspring have been generated (i.e., entirely replace the previous population). Each child is subjected to an operator, which imposes a constant mutation probability of 0.005. The offspring are evaluated, ranked, and proceed to become the parents of the next generation. For this experiment, we maintained a constant population size of 200 and ran the algorithm for 5,000 generations for a total of one million objective-function evaluations.

2.2.2 Hierarchical Bayesian optimization algorithm (hBOA)

HBOA used the same chromosomal encoding scheme as the SGA (Figure 3) but maintained a much larger population size of 5,000 individuals. In order to keep the total number of objective-function evaluations the same for both approaches, hBOA was allowed to run for 200 generations for a total of one million function evaluations. We used restricted tournament selection and set the tournament size to 12, based on empirical observations. We employed the same bit-wise mutation probability as with the SGA, 0.005. The maximum number of incoming edges for a single node in the Bayesian network was limited to four to avoid unnecessary linkage complexity. Elitism was adopted. In each generation, parental candidate solutions were evaluated and ranked, and those in the bottom half were replaced by newly generated offspring.

2.2.3 Objective Function

The objective function is a MATLAB subroutine, which was used by both the SGA and hBOA to evaluate candidate solutions. For each chromosome in the population, the subroutine extracts the 24 complex weights and computes the corresponding far-field radiation patterns for five discrete frequencies (9.0, 9.48, 10.0, 10.52, and 11.0 GHz). This experiment employed three variations of the objective function, which are described below as Cases 1, 2, and 3. (Note that for this experiment, we defined fitness such that *lower* values correspond to higher quality solutions. Traditionally, fitness is defined such that *higher* values correspond to higher quality solutions).

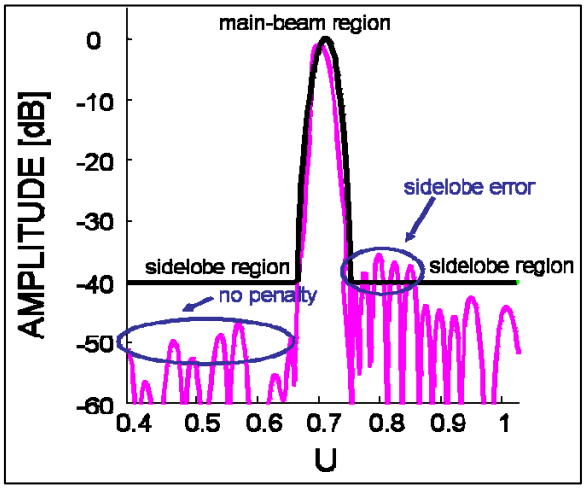

Figure 4. Case 1 objective function.

Figure 4 shows the objective function corresponding to Case 1. The pink curve is a typical far-field radiation pattern produced by the system at a specific frequency and for a given set of complex weights, w_i. The *x*-axis represents *u*-space (*i.e.*, $\sin\theta$), and the *y*-axis measures the normalized amplitude of the pattern in decibels. The black "mask" represents the objective function, which is divided into a *main-beam* region and a *sidelobe* region. For this case, we perform a point-by-point subtraction of the mask from the pattern. An error value E_k is computed by calculating the mean sum of the squared differences between the pattern and mask:

$$E_k(w, f_k) = \frac{1}{U}\sum_{i=1}^{U}\left[pattern_i - mask_i\right]^2 \ , \ \forall\left(pattern_i - mask_i \geq 0\right), \tag{3}$$

where $\boldsymbol{w}$ represents the vector of complex weights, f_k is the k^{th} discrete frequency, $pattern_i$ and $mask_i$ represent the i^{th} point of the radiated pattern and mask, respectively, and U represents the total number of points in the radiation pattern. Note that no penalty is administered when the pattern lies below the mask in the sidelobe region (*i.e.*, if the difference between the pattern and mask is negative, it is not used in the computation). In essence, we're trying to force the pattern to conform to the mask in the main-beam region while forcing the pattern to lie below the mask in the sidelobe region. Also note that we are "overshooting" by trying to force the algorithm to find a solution with –40 dB sidelobes in hopes that it will at least be able to obtain –30 dB. This lack of efficiency is an inherent weakness of this objective function. The overall Case 1 fitness value, $F_1(w)$, is the average of the error across the entire frequency band:

$$F_1(w) = \frac{1}{K}\sum_{k}^{K} E_k \,, \tag{4}$$

where K is the total number of discrete frequencies (and is equal to 5 for our experiment).

Figure 5 shows the objective function used for both Cases 2 and 3. For Case 2, the error has two components, the first of which is as follows:

$$E_{k,1}\left(w, f_k\right) = \left[1 - pattern\left(u_0\right)\right]^2 , \tag{5}$$

where u_0 is the desired steering angle of the pattern peak (u_0 = 0.7071 for this experiment). In essence, we need to ensure that the peak of the normalized pattern in the main-beam region coincides with the desired steering angle u_0. The second error component is:

$$E_{k,2}\left(w, f_k\right) = MSL^2 , \tag{6}$$

where *MSL* refers to the "maximum sidelobe level" (*i.e.*, the maximum level of the radiation pattern in the sidelobe region). In other words, we're trying to maximize the difference between the normalized pattern peak and the maximum sidelobe level as illustrated in Figure 5. The Case 2 fitness value, $F_2(w)$, is the mean summation of the error components across the entire frequency band:

$$F_2(w) = \frac{1}{K}\sum_{k=1}^{K}\left(E_{k,1} + E_{k,2}\right). \tag{7}$$

It is clear that the objective function for Case 2 involves only two subtractions, rather than a point-by-point comparison of the pattern to the mask. This property renders Case 2 more computationally efficient than Case 1. Similar to Case 1, however, the overall fitness value for a given set of complex weights is the average of the error across the entire frequency band.

Case 3 is identical to Case 2, except the overall fitness value, $F_3(w)$, is equal to the maximum error across frequency:

$$F_3(w) = \max_k \left(E_{k,1} + E_{k,2}\right). \tag{8}$$

In other words, Cases 1 and 2 are aimed at minimizing the *mean* error across frequency, whereas Case 3 minimizes the *maximum* error across frequency. Of the three objective functions, Case 3 is the most relevant to our particular problem, since we are ultimately trying to minimize the maximum sidelobe level across the entire frequency band.

2.3 Results: SGA vs. hBOA

We ran each case three times applying both the SGA and hBOA. Figure 6 shows the optimized far-field radiation patterns for the first run of each case at a frequency of 9.00 GHz (we chose to present this frequency, because it represents the worst-case system performance for both optimization approaches and for all three cases). In each plot, the optimal SGA solution is depicted by the blue solid line, and the optimal hBOA solution is represented by the red dotted line. Again, keep in mind that we defined fitness such that lower values correspond to higher quality solutions.

Figure 6 shows that the SGA and hBOA perform similarly when the Case 1 objective function is implemented. In particular, both approaches find solutions corresponding to a maximum sidelobe level of approximately -23 dB; thus, one may be tempted to assume that both algorithms perform equally well for this case. Yet, we found that the mean fitness across all three runs is 33% higher for the SGA when compared to hBOA, and the standard deviation of the fitness across all three runs for the SGA exceeds that for hBOA by a factor of 175! These results imply that, given enough runs, the best-quality SGA solution may be comparable to the best-quality hBOA solution; however, hBOA seems to be a much more consistent and reliable search mechanism.

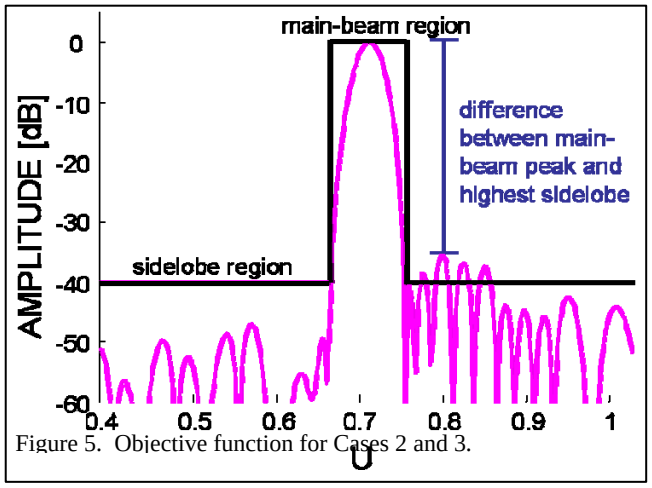

Figure 5. Objective function for Cases 2 and 3.

HBOA slightly outperforms the SGA when the Case 2 objective function is implemented. Here, we see that the peak sidelobes corresponding to the SGA and hBOA are -25 dB and -27.5 dB, respectively – a difference of 2.5 dB. For the Case 3 objective function, we see a huge difference in solution quality between the SGA and hBOA. Whereas the peak sidelobe of the hBOA solution is -28 dB, the radiation pattern corresponding to the best SGA solution doesn't have a well-defined main lobe and possesses sidelobes as high as -6 dB.

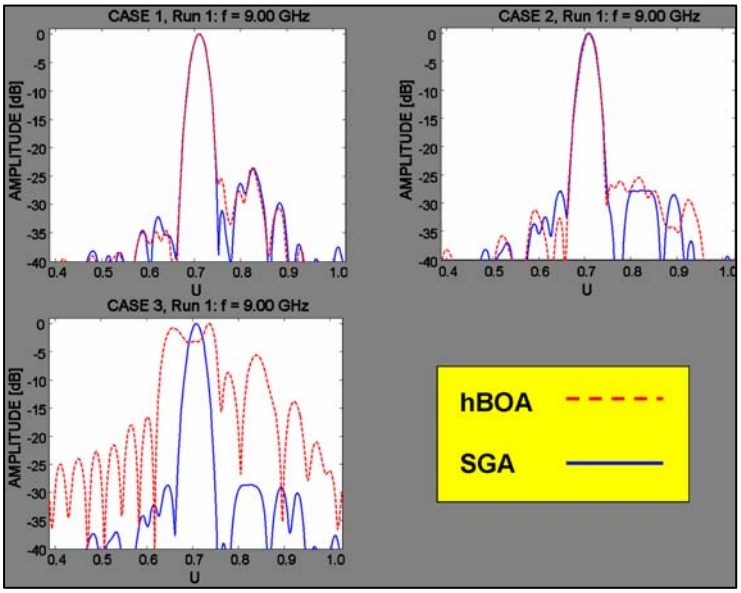

Figure 6. Optimization results.

Overall, these results are not surprising. The different objective functions represent drastically different solution spaces. Case 1 involves forcing a function to a mask, which is considered a GA-easy problem, because taking the average makes the fitness-function landscape smooth. Thus, we see that, in this case the SGA performs comparably to hBOA. Cases 2 and 3 are considered GA-difficult, because the min/max nature of the objectives give rise to a solution space that contains many local minima. The SGA, therefore, easily fell into some local minimum and was not capable of exploring the landscape globally. HBOA, on the other hand, was able to better identify the linkage of the problem, which allowed it to recombine salient pieces of information without disrupting good building blocks.

To conclude, for simple problems SGAs are preferred, since they are computationally inexpensive and the solution quality is comparable to that of competent GAs; however, for difficult problems competent GAs should be adopted because, based on our observations, these techniques are able to achieve higher-quality solutions than SGAs.

3 POLYOMINO SUBARRAYS

This section describes a method to measure the "degree of similarity" between two arbitrary planar array geometries. Basically, we borrow a concept from neural network theory, which we refer to as the "dot-product metric," to effectively categorize polyomino arrays based on their degree of similarity. Section 3.1 introduces the concept of a polyomino-subarray architecture for a planar array. Section 3.2 provides an overview of the dot-product metric and describes its application to this problem. Finally, Section 3.3 presents our results and conclusions.

3.1 Polyomino subarrays

Phased-array systems are typically designed to operate over a finite frequency band. For small arrays, time-delay units are often inserted behind each element to prevent the main beam of the radiated pattern from "squinting" as the frequency of operation strays from the center frequency. As the number of array elements increases, however, it becomes prohibitively expensive to insert time delay at the element level; thus, time delay is introduced by using phase shifters at the element level and time-delay units behind rectangular groups of elements, or *subarrays*. Thus, if we construct an array architecture, which consists of *N*-element subarrays, we can reduce the number of time-delay units by a factor of *N* – a substantial cost savings. Rectangular subarrays, however, come with a heavy price. Typically, one element in each subarray is chosen to be the "phase center," which simply means that it is the phase reference for all other elements within the subarray. Usually, all subarrays within the array have identical phase centers; thus, the periodic nature of the subarray phase centers in a rectangular-subarray architecture leads to significant quantization lobes in the radiated pattern. Obviously, quantization lobes represent severe pattern degradation.

There have been several recent papers describing the use of random or irregular-shaped subarrays (i.e., non-rectangular) to randomize the phase-center locations of the subarrays, which in turn, leads to quantization lobe suppression [12, 13]. In addition, we have shown that the use of polyomino-shaped subarrays can provide significant quantization lobe suppression and that the subarrays can be realized entirely in the control network that feeds the elements [14, 15, 16].

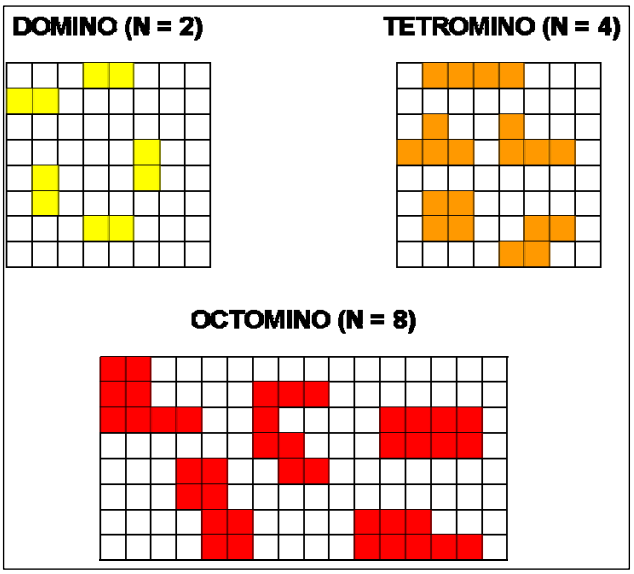

Figure 7. Examples of polyominos.

Polyominos are figures composed of elements on a square grid. As depicted in Figure 7, a polyomino of order *N* contains *N* square elements, where each element shares a common side with at least one other element (i.e., elements can be connected along the *x*- and *y*-dimensions only, not diagonally). Figure 7 shows examples of polyominos where *N* = 2 (domino), 4 (tetromino), and 8 (octomino). When every element of a rectangular grid belongs to one, and only one, polyomino shape, we say the grid is perfectly "tiled." Figure 8 shows a perfect tiling of a 32 x 32-element grid using L-shaped octominos. We can use this tiling to represent the subarray architecture of a 32 x 32-element antenna array.

We are currently conducting a trade-space study in which array performance characteristics are measured as a function of array size, subarray size, and polyomino shape. We use an automated tiling program [17] in conjunction with an electromagnetics simulation code to generate hundreds of arrays consisting of polyomino subarrays. In order to get meaningful results, we need to analyze a large number of arrays for each array type. For example, suppose we want to directly compare the performance of an array of L-shaped octomino subarrays with an array of C-shaped octomino subarrays. For each shape, we could generate 100 arrays and conduct a statistical analysis of array performance. Before we begin the analysis, however, we need a method to ensure that the 100 tilings for each shape represent a uniform sampling of the space of possible tilings for that shape and array size. For example, consider a 32 x 32-element array of 128 L-shaped octominos. There are literally millions of ways to perfectly tile the array. If we wish to perform a statistical analysis of array performance with respect to the tiling geometry, we need to use a set of tilings such that every tiling within the set has a significantly "different" geometry from every other tiling. The next section describes the metric we used to measure the degree of similarity between tilings.

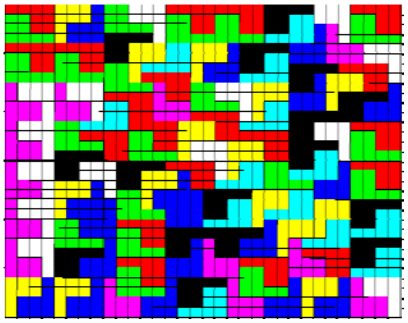
Figure 8. Perfect tiling of a 32 x 32 – element grid using L-shaped octominos.

3.2 Dot product metric

In order to measure the degree of similarity between two arbitrary tilings, we apply what we call the *dot-product metric*. This metric is implemented in neural network theory for pattern classification problems [18]. Basically, each pattern (i.e., tiling) is represented as a multi-dimensional vector. The angle between any two vectors **A** and **B** can be computed using the following:

$$\theta_{AB} = \cos^{-1}\left[\frac{\bar{A} \bullet \bar{B}}{\|\bar{A}\|\|\bar{B}\|}\right]. \tag{9}$$

Smaller values of θ correspond to patterns that have a high degree of similarity, whereas larger values of θ indicate dissimilar patterns.

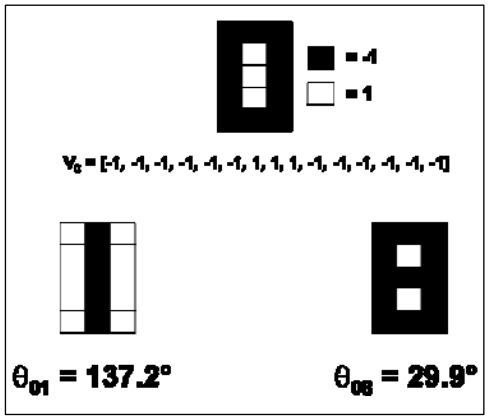

Figure 9. Dot-product-metric example.

To illustrate this point, consider the pattern classification problem of Figure 9. Here, a 3 x 5 grid is used to represent numerical values from 0 to 9 by blackening the appropriate blocks. The pattern which represents the number zero is shown at the top of the figure. If we assign each black box a value of -1 and each white box a value of 1, we can describe this pattern in vector format (by taking the values column-wise) as follows: V_0 = [-1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1]. The patterns representing the numerals one and eight are shown at the bottom of the figure. Applying equation (9), it can be shown that the dot-product angle between patterns zero and one, θ_{01}, is 137.2°, whereas θ_{08} is only 29.9°. These results are expected, since the "zero" pattern is very similar to the "eight" pattern (i.e., differs by only a single block-value), whereas the "zero" pattern is quite dissimilar to the "one" pattern (i.e., the two patterns have only two blocks in common).

Consider the L-shaped octomino. There exist eight orientations as shown in Figure 10. Each orientation is assigned a unique bipolar, binary representation; thus, all eight elements within the octomino are assigned the same binary number. Again, taking the array elements column-wise, a 3,072-element vector is formed (i.e., 1,024 elements x 3-bit representation) for each tiling. When the tiling shown at the top of the figure is compared to the tiling directly below, we see that the dot-product angle is only 10.1°. This result is expected, since these two tilings differ only in the small area indicated by the circles; otherwise, the tiling geometries are identical. On the other hand, the dot-product angle between the top tiling and the bottom-right tiling is much larger, indicating that the two patterns are dissimilar, and this dissimilarity is easily recognized by a simple visual comparison.

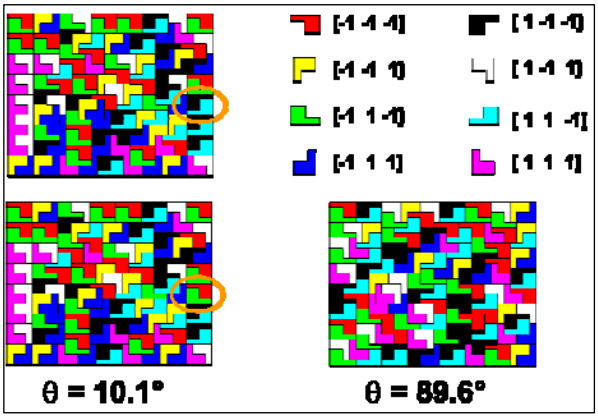

Figure 10. Dot-product metric used to determine degree of similarity among tilings of L-shaped octominos.

3.3 Results

We generated 99 32 x 32-element arrays of L-shaped octominos and computed the dot-product angle between each tiling and every other one. The values for θ ranged from roughly 75° to 105°, indicating that there is a fairly high degree of dissimilarity across the entire set of tilings. We subsequently computed the far-field radiation patterns of all 99 arrays and calculated the peak sidelobe level for each. Figure 11 is a histogram of peak sidelobe level for all 99 arrays at a normalized frequency of $f / f_0 = 0.75$, where f_0 represents the center frequency. The *x*-axis represents the amplitude of

the peak sidelobe level in decibels, and the *y*-axis represents the number of tilings contained in each amplitude bin. Note that the histogram is a crude approximation of a Gaussian distribution, which is what one would expect if the 99 tilings truly represent a random sampling of the space of possible tilings. In other words, if we had chosen a set of tilings which exhibit a high degree of similarity across the entire set (i.e., small dot-product angle), the array performance corresponding to each tiling would be very similar, and the histogram would consist of a single amplitude bin. This result lends support to the validity of the dot-product metric as an estimator of similarity among tilings.

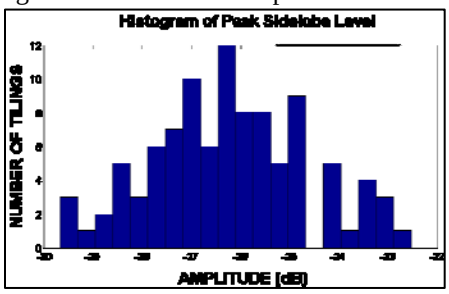

Figure 11. Array performance statistics.

4 CONCLUSION

This paper has described the application of biologically-inspired algorithms and concepts to the design of wideband antenna arrays. In particular, we have demonstrated the merits of applying competent GAs to difficult problems in which the solution space contains many local minima. HBOA was able to learn the linkage of the optimization problem and find an acceptable solution even when a computationally convenient, but GA-difficult, objective function was employed. The SGA, on the other hand, could only find a reasonably good solution when a more conventional, computationally expensive objective function was implemented. We have also demonstrated the validity of using the dot-product metric to measure the degree of similarity among a set of polyomino tilings. This allows us to sample the space of possible tilings uniformly, allowing us to calculate array performance characteristics as a function of tiling geometry.

ACKNOWLEDGEMENTS

This work was supported by the Air Force Office of Scientific Research, Mathematics and Space Sciences Directorate, under Dr. A. Nachman.

REFERENCES

1. Santarelli, S., Yu, T., Goldberg, D., Altshuler, E., O'Donnell, T., Southall, H., Mailloux, R. (2006), "Military Antenna Design Using Simple and Competent Genetic Algorithms, Journal of Mathematical and Computer Modeling, **43** (2006) 990 - 1022.
2. Pelikan, M. (2002). *Bayesian optimization algorithm: From single level to hierarchy*. Ph.D. thesis, University of Illinois at Urbana-Champaign, Urbana, IL. (Also IlliGAL Report No. 2002023).
3. Goldberg, D.E. (1991). Theory tutorial. (Tutorial presented with G. Liepens at the 1991 International Conference on Genetic Algorithms, La Jolla, CA).
4. Howard, R.A., & Matheson, J.E. (1981). Influence diagrams. In Howard, R. A., & Matheson, J. E. (Eds.), *Readings on the principles and applications of decision analysis*, Volume II, pp. 721-762, Menlo Park, CA: Strategic Decisions Group.
5. Simon, H.A. (1968). *The sciences of the artificial.* Cambridge, MA: The MIT Press.
6. Pelikan, M., & Goldberg, D.E. (2000), "Hierarchical problem solving by the Bayesian optimization algorithm," Genetic and Evolutionary Computation Conference (GECCO-2000), pp. 267–274. (Also IlliGAL Report No. 2000002).
7. Pelikan, M. & Goldberg D.E. (2003). Hierarchical BOA solves Ising spin glasses and MAXSAT. *Genetic and Evolutionary Computation Conference 2003 (GECCO-2003)*, Springer-Verlag, pp. 1271-1282 (Also IlliGAL Report No. 2003001).
8. Pelikan, M., Goldberg D.E., & Sastry, K. (2001), "Bayesian Optimization Algorithm, Decision Graphs, and Bayesian Networks," Genetic and Evolutionary Computation Conference 2001 (GECCO-2001), Springer-Verlag, pp. 519-526. (Also IlliGAL Report No. 2000020).

9. Pelikan, M., & Goldberg, D.E. (2001), "Escaping Hierarchical Traps with Competent Genetic Algorithms," Genetic and Evolutionary Computation Conference 2001 (GECCO-2001), Springer-Verlag, pp. 511-518. IlliGAL (Also IlliGAL Report No. 2001003).
10. Mailloux, R. J. (2001), "A low-sidelobe partially overlapped constrained feed network for time-delayed subarrays," IEEE Transactions on Antennas and Propagation, **Vol**. **49**, No 2, pp. 280–291.
11. Goldberg, D.E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.
12. R. C. Hansen, and G. G. Charlton, "Subarray Quantization Lobe Decollimation," IEEE Trans. **AP-47**, No.8, August 1999, pp.1237-1239.
13. V. Pierro, V. Galdi, G. Castaldi, I. M. Pinto, and L. B. Felson, "Radiation properties of planar antenna arrays based on certain categories of aperiodic tilings," IEEE Trans. **AP-53**, No.2, February 2005, pp. 635-643.
14. R. J. Mailloux, S. G. Santarelli, T. M. Roberts, "Irregular shaped subarrays for time delay control of planar arrays," *Proceedings of the 2004 Antenna Applications Symposium*, Monticello, Illinois.
15. R. J. Mailloux, S. G. Santarelli, T. M. Roberts, "Polyomino shaped subarrays for limited field of view and time delay control of planar arrays," *Proceedings of the 2005 Antenna Applications Symposium*, Monticello, Illinois.
16. R. J. Mailloux, S. G. Santarelli, T. M. Roberts, "New results using polyomino-tiled subarrays for time-delay control of wideband arrays," *Proceedings of the 2006 Antenna Applications Symposium*, Monticello, Illinois.
17. S. Montgomery-Smith, "polyomino-0.4," available online http://www.math.missouri.edu/_~stephen/software/polyomino
18. M. T. Hagan, H. B. Demuth, M. H. Beale, *Neural Network Design*, Martin Hagan Publishing, 2002.